ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

COST-EFFECTIVE AUTONOMOUS INDUSTRIAL LOGISTICS ROBOT

K. Thangaraj, R. Ragupathy, C. Adrien Periyanayagam, J. Muruganandham and D. Meena
Department of Electrical and Electronics Engineering, Sri Manakula Vinayagar Engineering College, Puducherry, India
E-Mail: thangaraj.electres@gmail.com

ABSTRACT

An autonomous logistics robot is designed for the purpose of transporting goods from one location to another without the need for human intervention. In previous models, robots relied on magnetic strips to navigate and reach their destinations. However, these conventional models had some drawbacks; if there were obstacles in their path, the robots would become stuck and unable to proceed towards their destination. To address these issues, a new model has been proposed. This model eliminates the necessity for human intervention and magnetic strips to guide the robot to its destination. Moreover, the cost of this model is significantly lower than that of the conventional approach. Consequently, this proposed model establishes the logistics robot as fully autonomous, eliminating the requirement for human interference. The robot achieves autonomous movement through the utilization of LiDAR Depth sensors and obstacle sensors. The Robotic Operating System (ROS) in conjunction with Raspberry Pi 3 is employed to map the path for the autonomous robot.

Keywords: ROS, SLAM, autonomous navigation, path mapping, self-localization, LiDAR.

Manuscript Received 30 January 2024; Revised 17 May 2024; Published 30 June 2024

1. INTRODUCTION

In Industries, the transportation of goods necessitates their relocation from one location to another. Historically, forklifts were employed for this purpose, requiring manual labor and posing a potential for inaccuracies. To address these challenges, the concept of an autonomous robot capable of independently transporting goods emerged. This robot would operate devoid of human intervention, navigating its route seamlessly and efficiently. It would possess the ability to detect and circumvent obstacles en route to its designated endpoint. The proposed model outlines the development of a fully autonomous logistics robot, designed to traverse from the source to the destination autonomously. Logistics robots streamline the storage and movement of goods within warehouses and storage facilities, constituting an integral aspect of intralogistics [1], [2]. Beyond these environments, their applications extend to various contexts. The implementation of logistics robots promises enhanced operational efficiency compared to manual labor, leading to heightened productivity and profitability for adopters.

These robots function within predetermined routes, facilitating the continuous movement of items for shipment and warehousing. The integration of LiDAR and depth sensors significantly contributes to cost reduction within logistics, streamlining supply chain operations.

Logistics robots come in diverse deployment forms, yet their commonality lies in being mobile units designed to mechanize the movement of merchandise [3], [4], [5]. Their substantial uptime remains a prime factor influencing their profitability, independent of the specific application. Enterprises are recognizing the inherent advantages of incorporating logistics robots, resulting in a poised expansion of the market.

SLAM algorithms offer the potential to enact self-guided navigation. Through SLAM, a robot's positioning is estimated as it traverses unfamiliar terrain. The frequency of exploration attempts directly correlates with the quality of the resultant map. The execution of SLAM algorithms is facilitated by utilizing tools such as GMapping. This tool necessitates two primary data inputs: odometry data stemming from wheel encoders and LiDAR data. With Slam Gmapping, a 2-D occupancy grid map akin to architectural floor plans is generated. The robotic software development takes place within ROS (Robot Operating System), a platform that furnishes services encompassing inter-process communication, packet messaging, device control, and fundamental feature implementation. In ROS, processing entities are depicted graphically, and computation is distributed across nodes [6], [7], [8].

For intricate 3D simulations, the GAZEBO environment is harnessed, composed of key constituents: Models, GZ Server, GZ Client, and GZ Web. GAZEBO introduces an integrated robot model, cloud-based simulation, dynamics emulation, advanced 3D visuals, command line utilities, TCP-IP communication, among others. GAZEBO's capabilities span the visualization of inertia, forces, and sensor-related data [9],[10].

ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

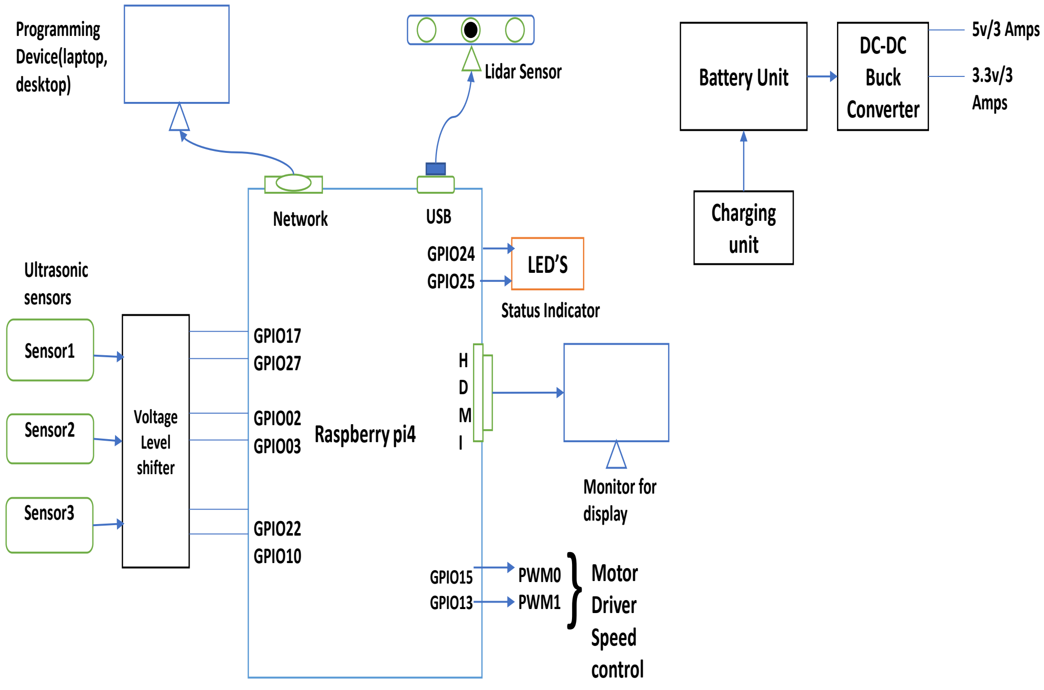

Figure-1. Block diagram of industrial logistics robot.

Logistics robots streamline the storage and movement of goods as they navigate the supply chain. They find application predominantly within warehouses and storage facilities, organizing the arrangement and conveyance of items - a process termed intralogistics [11], [12], [13]. However, their versatility extends beyond these environments. Embracing logistics robots yields substantial increases in operational efficiency compared to manual labor, resulting in significant enhancements in productivity and financial gains for adopters.

These robots adhere to predetermined paths, organizing the transportation of products for both shipment and warehousing throughout all hours. A pivotal role in optimizing logistics costs and refining supply chain operations is played by LiDAR and Depth sensors. The proposed model incorporates key elements, including the LiDAR Sensor, ultrasonic sensors, encoded motors, display device, indication lights, Raspberry Pi 3, and buck converters as shown in Figure-1.

2. SLAM ALGORITHM

Autonomous robots are capable of navigating both indoor and outdoor environments while avoiding collisions with obstacles. Concurrent Localization and Mapping (SLAM) is a technique employed to simultaneously generate and update maps while estimating the robot's position. Several algorithms are utilized to address this challenge, including Particle Filter, Extended Kalman Filter, Fast SLAM, Covariance Crossing, and Graph-based SLAM. SLAM algorithms encompass various aspects such as mapping, kinematic modeling, handling multiple sensors, moving objects, loop closure, search, and dealing with complexity. An essential element of SLAM is the rangefinder, which observes the surroundings, and the accuracy of distance measuring devices depends on numerous variables [14], [15,] [16]. Robots utilize these measuring devices and sensors to establish their location based on landmarks. Once the robot identifies landmarks, it processes the input data to recognize its environment.

Among the available SLAM algorithms are Core SLAM, Gmapping, KartoSLAM, Lago SLAM, and Hector SLAM. In this context, the chosen SLAM algorithm is Gmapping, which relies on particle filters. The primary objective of the particle filter is to compute the posterior distribution of the states in a Markov process, even in the presence of noisy partial observations. This approach employs approximate predictions and updates, representing a sample from a distribution with particles. Each particle is associated with a weight indicating the probability of that particle being sampled. Gmapping, specifically, is a highly capable Rao-Blackwellized particle filter that generates grid maps from LiDAR data [17], [18].

A. Map Formation

Map formation begins with the initial step of terrain generation and the subsequent integration of a turtle bot within Gazebo. The robot model, in this case, comprises two buses, housing an internally mounted detector. The mapping process is facilitated through the utilization of the gmapping package. To successfully employ gmapping, the robot model necessitates the provision of odometry data and the presence of a horizontally fixed LiDAR rangefinder. This rangefinder captures depth images of the terrain through the use of infrared detectors, serving as a range finding device. These depth images are then converted into a 3D point cloud

ARPN Journal of Engineering and Applied Sciences

utilizing the depth image proc package within the ROS framework. Subsequently, this 3D point cloud data is transformed into a 2D LiDAR scan employing the point cloud to LiDAR scan package. By supplying the requisite parameters to the packages within gmapping, a chart is generated in rviz. Initially, the robot's movement within the simulated terrain is achieved using the teleop crucial package in ROS, with velocity commands transmitted via the keyboard. The gmapping package yields chart data, which can be saved by the chart garçon package, preserving the generated chart for future use.

B. Localization

Localization involves determining the robot's position and orientation relative to its surroundings. Key components for achieving this include odometry data and environmental maps, which are crucial for interpreting LiDAR data collected by the robot. In the context of Turtlebot, localization is accomplished using AMCL (Adaptive Monte Carlo Localization), which adopts a probabilistic approach to localize robots in a two-dimensional space. AMCL employs particle filters to continually track the robot's pose relative to a predefined map of the environment.

C. Rqt Graph

The ROS (Robot Operating System) offers a GUI plugin called Rqt-Graph as part of the Rqt tool suite. This invaluable tool serves the purpose of visually representing all active nodes and processes, as well as illustrating the communication pathways that connect them. The Rqt graph is particularly adept at offering a comprehensive overview of the system under consideration, making it a valuable asset for system debugging and analysis.

D. Autonomous Navigation

Autonomous navigation becomes possible once the mapping and localization processes have been successfully executed. To achieve this, ROS navigation packages are leveraged to implement AMCL (Adaptive Monte Carlo Localization). The responsibility of orchestrating localization on a static map is delegated to a ROS node within the AMCL package. This node subscribes to various data sources, including TF data, 2D LiDAR scan data provided by the robot, and the initial static map. Subsequently, the AMCL node disseminates information regarding the robot's pose and its estimated position relative to the map [19], [20].

Critical information about obstacles in the simulated environment is stored in two separate cost maps. The global cost map plays a pivotal role in global path planning, enabling the creation of long-term route plans across the map. On the other hand, the local cost map is indispensable for obstacle avoidance and local path planning. Rviz serves as a visualization tool, offering a clear representation of both the global and local cost maps. With these foundational components in place, the robot gains the capability to navigate autonomously. Subsequently, the robot autonomously plans its path to reach the specified destination, and it executes its movements accordingly.

3. HARDWARE IMPLEMENTATION

A. Hardware Components

The Raspberry Pi 3 serves as the central control unit for this industrial logistics robot, overseeing its entire operation. For environmental perception, the robot is equipped with a LiDAR -Depth sensor, which includes LiDAR sensors and infrared projectors and detectors. These sensors employ structured light or time-of-flight calculations to capture depth information, enabling real-time gesture recognition and body-skeletal detection. Additionally, the robot features a display device for user interaction.

To navigate and avoid obstacles, the robot relies on three Ultrasonic sensors. These sensors emit ultrasonic sound waves to measure the distance to objects in the robot's path, facilitating obstacle avoidance and safe navigation.

The robot's mobility is powered by two Geared encoded motors. These Geared DC Motors are designed with a gear assembly attached to the motor, allowing for precise control over speed and torque. The speed is typically measured in RPM (rotations per minute), and the gear assembly plays a crucial role in adjusting the motor's speed as needed. This principle of using gears to reduce speed while increasing torque is known as gear reduction.
The 12V, 30:1 Gear Motor with Encoder is a robust motor designed to operate on either a 12V or 5V power supply. It features a durable metal gearbox with a 30:1 gear ratio and comes equipped with an integrated quadrature encoder. This encoder offers a resolution of 64 counts for every revolution of the motor shaft, translating to 1920 counts for each revolution of the gearbox's output shaft.

One of the key components in the robot's power system is a 12V Li-ion battery (Lithium-Ion). Li-ion batteries are widely used in various applications due to their lightweight nature, high capacity, and efficient power delivery. They are commonly found in drones, radio-controlled aircraft, cars, and model trains.

To regulate the voltage from the 12V Li-ion battery, a DC-DC Buck Converter is employed. This device steps down the voltage to the required level, ensuring that the robot's components receive the appropriate power supply.

B. Hardware Implementation

The primary hardware components are integrated with a Raspberry Pi 3 (with 2 GB RAM) serving as the controlling device to monitor and manage the activities of these components. In the control system, the motors and sensors are actively enabled. There are two types of sensors utilized: ultrasonic sensors positioned in various directions and a LiDAR Depth Sensor placed at the front of the robot. These sensors are connected to the Raspberry Pi. Additionally, two 5kgCm stall torque motors are employed for movement, and a BTS7960 Motor Driver is used to control them with a power supply. An IM 2956

Buck converter is implemented to step down the voltage from its input (supply) to the required output (load). Power is provided by a 3.7V 2600mAH Lithium-ion Battery, which is managed by a Battery Management System (BMS 3S) to ensure proper power distribution to each component.

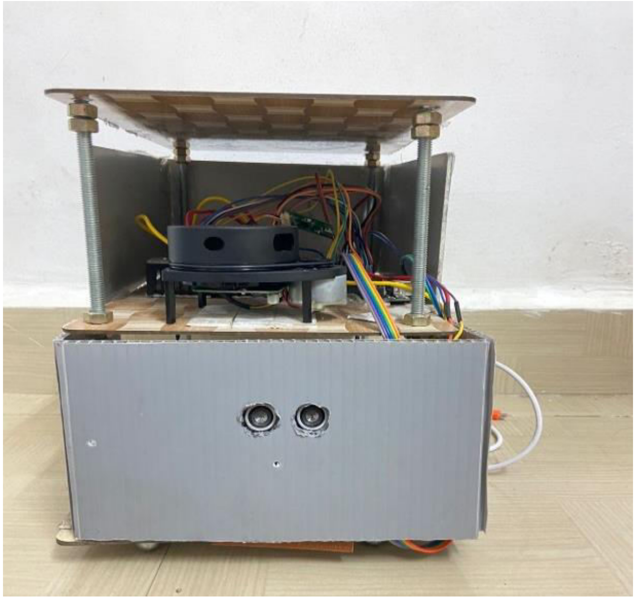

Figure-2. Hardware implementation.

The LiDAR Depth Sensor is a type of depth-determining sensor that provides both depth (D) and color (LiDAR) data in real time. The LiDAR's color coding represents various range limits: Red for short range, Green for distances above Red's limit, and Blue for long-range measurements. It is also referred to as a 3D depth sensor.
The Raspberry Pi 3 serves as the control center, executing software programs (instructions) to operate and manage the hardware components seamlessly.

The BTS7960 Motor Driver is a fully integrated high-current half-bridge driver designed for electrical motors, with an operating voltage of 24Vdc. The BMS 3S is a Battery Management System for a 3S battery pack, ensuring the safety and optimal performance of the battery pack. In this system, the motors are connected to the motor driver, the battery is connected to the BMS 3S circuit, and the sensors are directly interfaced with the Raspberry Pi 3.
The overall operation of the robot involves the collection of data from its environment using the LiDAR Depth sensor, which is then used to create a map with the assistance of SLAM (Simultaneous Localization and Mapping) tools. Mapping is carried out according to predefined requirements. When tasked with transporting a load, the robot autonomously navigates from its starting point to its destination. In the event of encountering an obstacle along the way, it performs obstacle avoidance maneuvers to safely reach its destination. Throughout the journey, the robot's position is continuously monitored and captured, allowing for precise tracking.

Figure-2 provides an overview of the hardware setup for this industrial logistics robot, which can autonomously navigate and avoid obstacles without requiring human intervention.

4. EXPERIMENTAL RESULTS ANALYSIS

A. LiDAR Sensor Results

The Gmapping software processes data from the LiDAR data acquisition and processing package. A visualization of this data object is logged, as depicted in Figure-3. The communication data includes parameters such as angle increment, minimum angle, maximum angle, range minimum, range maximum, and time per scan. This results in the generation of two 1081x1 matrices: one for range data and one for laser intensity.


```
header:
  seq: 814
  stamp:
    secs: 1535270563
    nsecs: 206067892
  frame_id: /Lidar_frame
angle_min: 0.0
angle_max: 3.14159274101
angle_increment: 0.0349065847695
time_increment: 0.0
scan_time: 450.0
range_min: 0.0
range_max: 100.0
ranges: [1.15619997300212e-19, 7.436256547647864e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374
e-40, 7.444328026802374e-40, 0.0, 0.0, 0.0, 0.0, 7.444356052771661e-40, 7.444328026802374e-40, 0.0, 7.444412104710234e-40, 7.444468156648807
e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-
40, 7.444328026802374e-40, 7.444328026802374e-40, 7.417983615673068e-40, 7.444328026802374e-40, 0.0, 7.441189118242287e-40, 7.44197384538230
9e-40, 7.44275857252233e-40, 7.443543299662353e-40, 0.0, 0.0, 7.418263875365933e-40, 7.418432031181652e-40, 7.444328026802374e-40, 7.4186001
8699737e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 0.0, 7.444328026802374e-40, 7.41209
8162122904e-40, 7.444328026802374e-40, 7.411873954368612e-40, 7.411201331105736e-40, 7.411425538860028e-40, 7.41164974661432e-40, 7.44432802
6802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.4443280268
02374e-40, 7.440964910487995e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 7.444328026802374e-40, 35050560.0, -0
.00017115524678956717, -2.1433956831629075e-09, -2.777443239281335e+31, 7.020209066843641e-18, -0.035263191908597946, 1.143787610922879e-19,
 0.0, 7.776982269248443e-40, 8828168.0, 137924.421875, -2.777505619853627e+31, 137728.0, -2.145043920265266e-09, -1.910565261553643e-09, -0.
03327140212059021, 0.0, 7.776982269248443e-40, 1.1437881279107618e-19, 7.776982269248443e-40, 0.0, 1.7257784419584495e+29, 7.482587984324162
e-33, -1.1762535548690256e-33, 1.2684756625056123e+35, 4.96568537827015e+24, 4.965722271758297e+24, 1.0054027800931877e+30, 1.38189937923227
27e+30]
```


Figure-3. Laser data script print output.

A direct screenshot of the parameter comments from the LiDAR scan script file, including definitions for each variable, is displayed in Figure-3. Among the notable parameters are angle min/max, time increment, and ranges.

VOL. 19, NO. 8, APRIL 2024 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

Angle min and max represent the angular ranges within which the sensor receives pulse readings, with the minimum angle origin assumed as 90° or π/2 radians when the robot is facing North.

The time measurements correspond to the intervals between each measurement, aligning with localization and odometry relative to the measurements taken.

Finally, the ranges denote the raw data measurements obtained from the LiDAR pulse inputs. These respective parameters are input into the Gmapping package, which then appropriately constructs and modifies the 2D map.

B. Results of Laser Scan of the Surrounding Area

A LiDAR sensor with a 180-degree rotating servo motor is mounted on the front of the mobile robot for mapping purposes. Data from the LiDAR sensor is transmitted to the computer via a USB 2.0 serial cable. The Rviz tool is employed to generate a map of the robot's surroundings with the assistance of the LiDAR sensor. Figure-4 illustrates the laser scan map of the experimental environment.

This map displays the laser scan of an enclosed room, with red marks within the map indicating obstacles, excluding the corners. It is a live map that continuously captures the real-time movement of objects. The laser data script is processed to create this type of map. The test environment encompasses dimensions of 3x4 meters, as depicted in Figure-4.

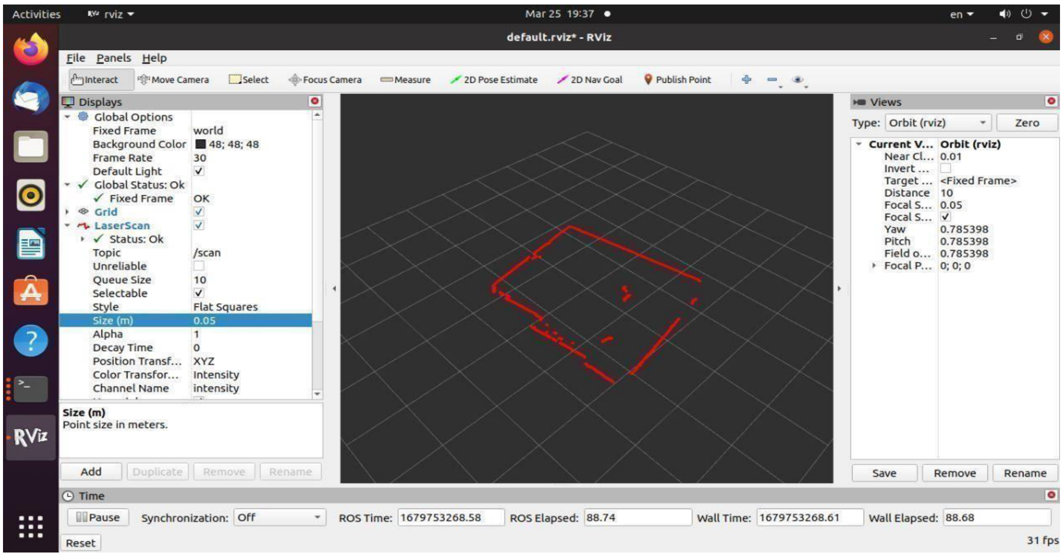

Figure-4. Laser scan map of the surrounding environment.

C. Results of Localization of the Autonomous Robot Using Rviz Visualization

Localization is the process of determining a mobile robot's position relative to its environment. This skill is fundamental for autonomous robots because knowing the robot's location is a crucial prerequisite for making decisions about its future actions. Figure-5 illustrates a typical robot localization scenario, where a map of the environment is accessible, and the robot is equipped with a LiDAR sensor to observe the surroundings and monitor its own motion.

The robot calculates its position and orientation within the map by utilizing data collected from the LiDAR sensors.

VOL. 19, NO. 8, APRIL 2024 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

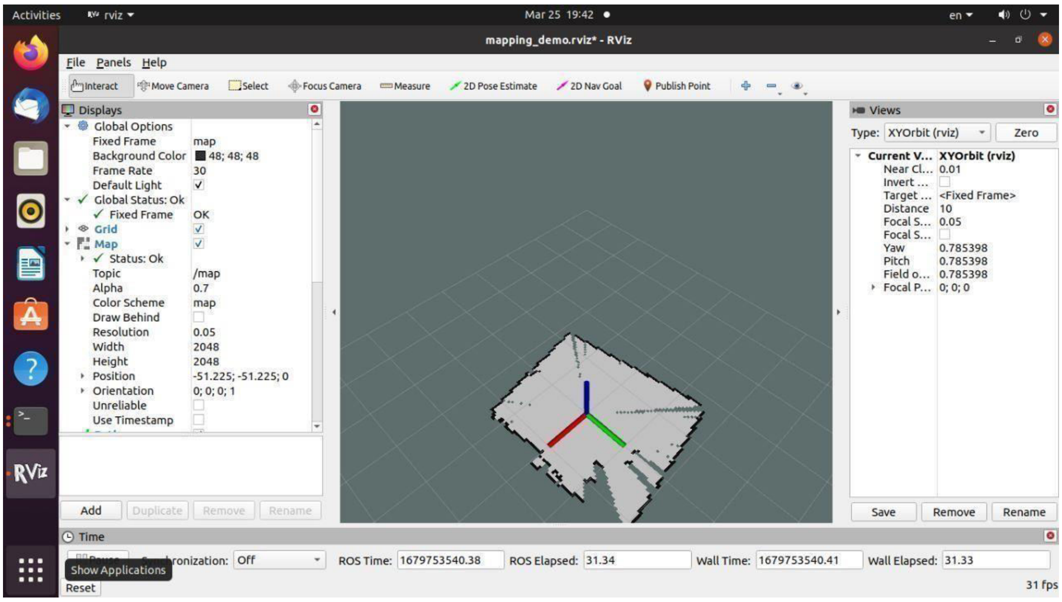

Figure-5. Localization of the autonomous robot in Rviz.

D. Results of Mapping of Autonomous Robot

The autonomous robot has been successfully mapped in the 2D map using the LiDAR sensor and the Rviz tool, as depicted in Figure-6. In the figure, the X, Y, and Z coordinates of the robot are indicated in red, blue, and green colors, respectively. The robot follows a predefined path from one location to another, which was previously mapped in the 2D map using the Rviz tool. The green line represents the designated path for the autonomous robot, which it adheres to once online to reach its destination.

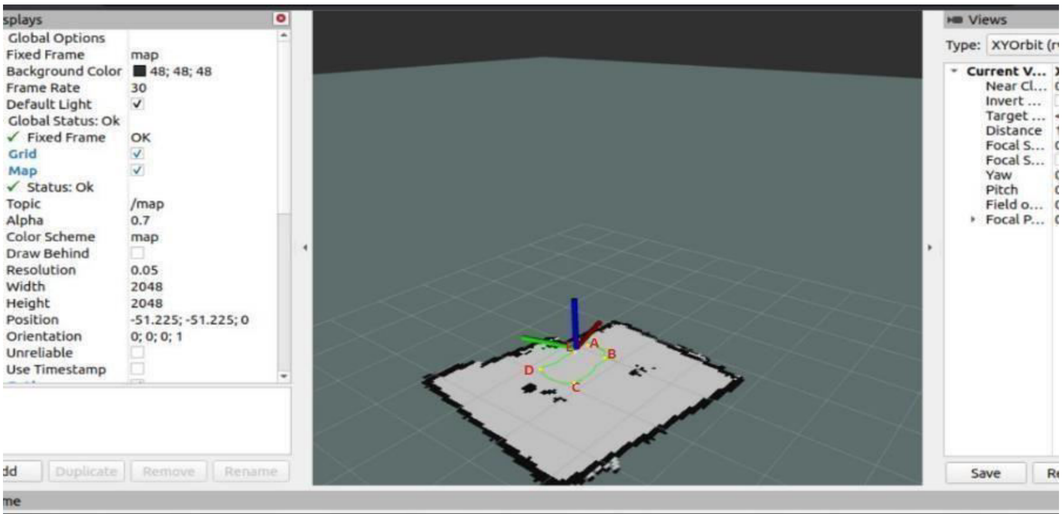

Figure-6. Actual snapshot of LiDAR 2D mapping at ~position (X, Y).

In case any obstacles obstruct its path, the robot automatically detects and avoids them with the assistance of ultrasonic sensors, ensuring it continues along its path to reach the destination. During the mapping process of the autonomous robot, five publish points are marked.

All the mapping points are represented in yellow color. Furthermore, for clarity and ease of understanding, these mapping points are labeled with alphabetical letters, starting with 'A' and followed by 'B', 'C', 'D', and ending with 'E' to indicate the starting and ending points.

To provide a clearer understanding, Figure-7 illustrates the sequential movement of the autonomous robot from point to point until it reaches its final destination at point 'E'.

ARPN Journal of Engineering and Applied Sciences

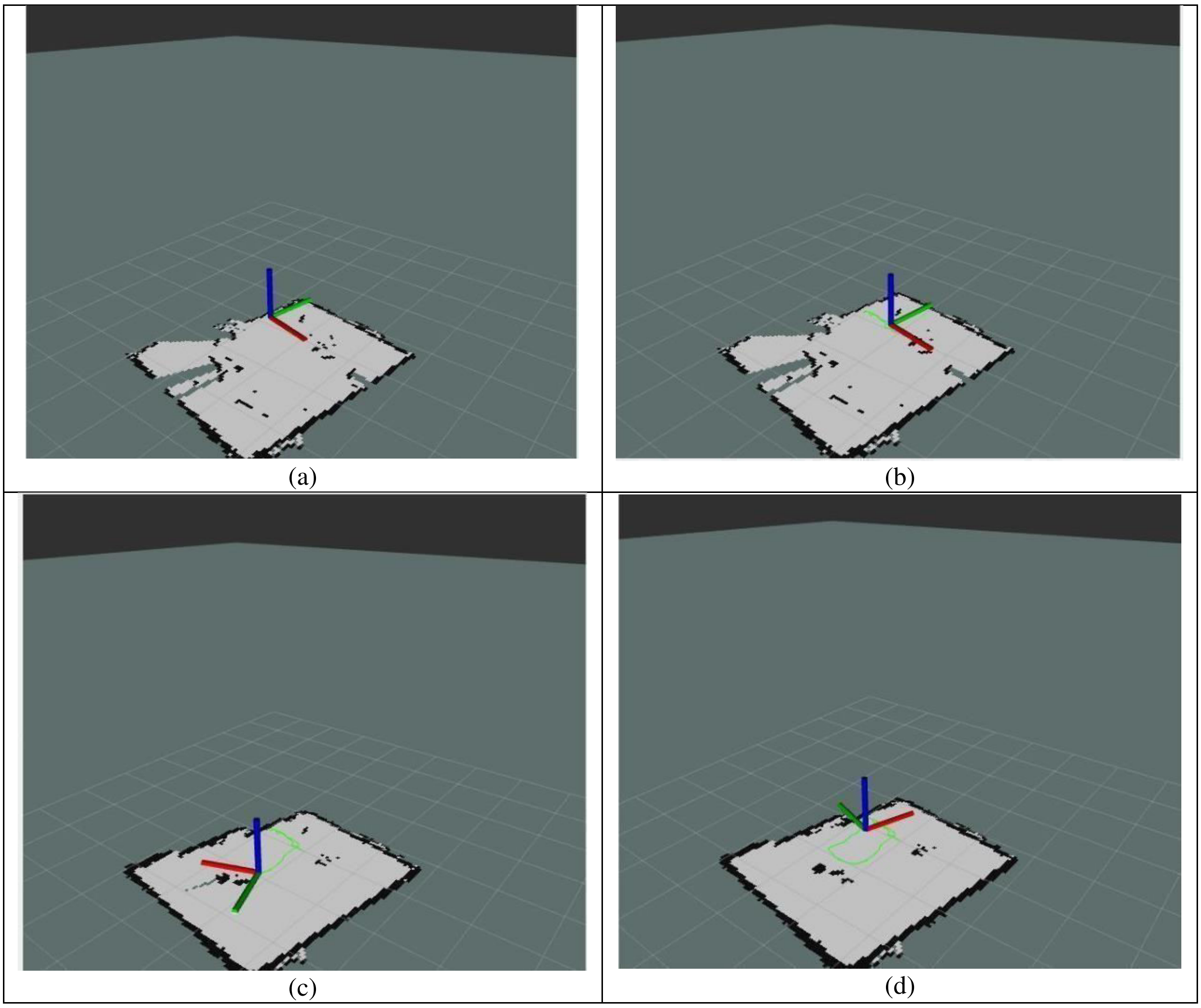

Figure-7. Autonomous robot movement from start point to end point.

In Figure-7, the movement of the autonomous robot is clearly evident as it navigates from its initial position to its designated destination. In Figure-7-A, the robot starts from the initial point within the 2D mapping area. Subsequently, in Figure-7-B, the robot successfully reaches its next programmed waypoint. Moving forward to Figure-7-C, the robot arrives at its third mapping point, and finally, in Figure-7-D, it successfully reaches its ultimate destination. This demonstrates the functionality of the autonomous robot navigation system.

5. CONCLUSIONS

This paper introduces a novel autonomous logistics robot designed to efficiently transport goods from one location to another without human intervention. It eliminates the need for both human intervention and costly magnetic strip infrastructure. The integration of LiDAR depth sensors and obstacle sensors enables the robot to autonomously navigate its environment, ensuring safe and efficient movement. By harnessing the power of the Robotic Operating System (ROS) in conjunction with the Raspberry Pi 3 as the central control unit, this model achieves autonomous path mapping and navigation using SLAM algorithm. Not only does this innovation enhance the robot's capabilities, but it also significantly reduces overall costs, making it a practical and cost-effective solution for logistics automation. In essence, this research establishes a paradigm shift in logistics robotics, presenting a fully autonomous solution that can adapt to dynamic environments and operate seamlessly, marking a promising advancement in the field of autonomous robotics.

REFERENCES

[1] Hengliang Tang, Anqi Wang, Fei Xue , Jiaxin Yang, And Yang Cao. 2021. A Novel Hierarchical Soft Actor-Critic Algorithm for Multi-Logistics Robots Task Allocation. *IEEE access,* doi: 10.1109/ACCESS.2021.3062457

[2] Jianhua Zhang, Ruyu Liu, Kejie Yin, Zengyuan Wang, Mengping Gui, Shengyong Chen. 2019. Intelligent Collaborative Localization among Air-Ground Robots for Industrial Environment Perception. IEEE Transactions on Industrial Electronics, 66(12), doi: 10.1109/TIE.2018.2880727

[3] Bernardo R., Sousa J. M. and Gonçalves P. J. 2022. Survey on robotic systems for internal logistics. Journal of Manufacturing Systems, 65, 339-350. https://doi.org/10.1016/j.jmsy.2022.09.014

[4] Nwankwo L., Fritze C., Bartsch K. and Rueckert E. 2023. ROMR: A ROS-based open-source mobile robot. HardwareX, 14, e00426. https://doi.org/10.1016/j.ohx.2023.e00426

[5] Wong S. and Yu Z. 2019. A ROS-Matlab road condition prediction algorithm with cost-effectiveness for self-navigating mobile robots. Procedia Manufacturing, 51, 19-25. https://doi.org/10.1016/j.promfg.2020.10.004

[6] Kis K. B., Csempesz J. and Csáji B. C. 2020. A simultaneous localization and mapping algorithm for sensors with low sampling rate and its application to autonomous mobile robots. *Procedia Manufacturing*, *54*, 154-159. https://doi.org/10.1016/j.promfg.2021.07.023

[7] Gadekar A., Fulsundar S., Deshmukh P., Aher J., Kataria K., Patel D. V. and Barve D. S. 2022. Rakshak: A modular unmanned ground vehicle for surveillance and logistics operations. *Cognitive Robotics*, *3*, 23-33. https://doi.org/10.1016/j.cogr.2023.02.001

[8] Seder M., Petrović L., Peršić J., Popović G., Petković T., Šelek A., Bićanić B., Cvišić I., Josić D., Marković I., Petrović I. and Muhammad A. 2018. Open Platform Based Mobile Robot Control for Automation in Manufacturing Logistics. IFAC-PapersOnLine, 52(22): 95-100. https://doi.org/10.1016/j.ifacol.2019.11.055

[9] SONUGÜR G. 2023. A Review of quadrotor UAV: Control and SLAM methodologies ranging from conventional to innovative approaches. Robotics and Autonomous Systems, 161, 104342. https://doi.org/10.1016/j.robot.2022.104342

[10] Perez-Grau F. J., Martinez-de Dios J. R., Paneque J. L., Acevedo J. J., Torres-González A., Viguria A., Astorga J. R. and Ollero A. 2021. Introducing autonomous aerial robots in industrial manufacturing. Journal of Manufacturing Systems, 60, 312-324. https://doi.org/10.1016/j.jmsy.2021.06.008

[11] Rohacz A., Weißenfels S. and Strassburger S. 2019. Concept for the comparison of intralogistics designs with real factory layout using augmented reality, SLAM and marker-based tracking. Procedia CIRP, 93, 341-346. https://doi.org/10.1016/j.procir.2020.03.039

[12] Raudmäe R., Schumann S., Vunder V., Oidekivi M., Nigol M. K., Valner R., Masnavi H., Singh A. K., Aabloo A. and Kruusamäe K. 2023. ROBOTONT - Open-source and ROS-supported omnidirectional mobile robot for education and research. *HardwareX*, *14*, e00436. https://doi.org/10.1016/j.ohx.2023.e00436

[13] Yilmaz A., Sumer E. and Temeltas H. 2022. A precise scan matching based localization method for an autonomously guided vehicle in smart factories. *Robotics and Computer-Integrated Manufacturing*, *75*, 102302. https://doi.org/10.1016/j.rcim.2021.102302

[14] Syrmos E., Bechtsis D. and Tsolakis N. 2021. MIROR: A middleware software tool for interfacing mobile industrial robots with optimization routing algorithms. *SoftwareX*, *17*, 100903. https://doi.org/10.1016/j.softx.2021.100903

[15] Lee W. J., Kwag S. I. and Ko Y. D. 2020. Optimal capacity and operation design of a robot logistics system for the hotel industry. *Tourism Management*, *76*, 103971. https://doi.org/10.1016/j.tourman.2019.103971

[16] Rohacz A., Weißenfels S. and Strassburger S. 2019. Concept for the comparison of intralogistics designs with real factory layout using augmented reality, SLAM and marker-based tracking. *Procedia CIRP*, *93*, 341-346. https://doi.org/10.1016/j.procir.2020.03.039

[17] Duchoň F., Babinec A., Kajan M., Beňo P., Florek M., Fico T. and Jurišica L. 2013. Path Planning with Modified a Star Algorithm for a Mobile Robot. Procedia Engineering, 96, 59-69. https://doi.org/10.1016/j.proeng.2014.12.098

[18] Yaacoub J. A., Noura H. N. and Piranda B. 2023. The internet of modular robotic things: Issues, limitations, challenges, & solutions. *Internet of Things*, *23*, 100886. https://doi.org/10.1016/j.iot.2023.100886

[19] Cox J., Tsagkopoulos N., Rozsypálek Z., Krajník T., Sklar E. and Hanheide M. 2023. Visual teach and generalise (VTAG)-Exploiting perceptual aliasing for

scalable autonomous robotic navigation in horticultural environments. Computers and Electronics in Agriculture, 212, 108054. https://doi.org/10.1016/j.compag.2023.108054

[20] Mello R. C., Sierra M., S. D., Scheidegger W. M., Múnera M. C., Cifuentes C. A., Ribeiro M. R. and Frizera-Neto A. 2022. The PoundCloud framework for ROS-based cloud robotics: Case studies on autonomous navigation and human-robot interaction. Robotics and Autonomous Systems, 150, 103981. https://doi.org/10.1016/j.robot.2021.103981